

AEDC-TR-90-32

Volume II

C.3

PA

**Application of System Identification
Techniques to Turbine Engine
Post-Stall Test and Evaluation**

**Robert P. Anex
Systems Control Technology, Inc.
Advanced Technology Division
Palo Alto, California 94303**

December 1990

Final Report for Period March 1988 through November 1989

**TECHNICAL REPORTS
FILE COPY**

**PROPERTY OF U.S. AIR FORCE
AEDC TECHNICAL LIBRARY**

Approved for public release; distribution is unlimited.

**ARNOLD ENGINEERING DEVELOPMENT CENTER
ARNOLD AIR FORCE BASE, TENNESSEE
AIR FORCE SYSTEMS COMMAND
UNITED STATES AIR FORCE**

NOTICES

When U. S. Government drawings, specifications, or other data are used for any purpose other than a definitely related Government procurement operation, the Government thereby incurs no responsibility nor any obligation whatsoever, and the fact that the Government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise, or in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.


Qualified users may obtain copies of this report from the Defense Technical Information Center.

References to named commercial products in this report are not to be considered in any sense as an endorsement of the product by the United States Air Force or the Government.

This report has been reviewed by the Office of Public Affairs (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations.

APPROVAL STATEMENT

This report has been reviewed and approved.



MARJORIE S. COLLIER
Directorate of Technology
Deputy for Operations

Approved for publication:

FOR THE COMMANDER



KEITH L. KUSHMAN
Technical Director
Directorate of Technology
Deputy for Operations

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.				
1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE December 1990	3. REPORT TYPE AND DATES COVERED Final, March 1988 -- November 1989		
4. TITLE AND SUBTITLE Application of System Identification Techniques to Turbine Engine Post-Stall Test and Evaluation, Volume II		5. FUNDING NUMBERS F40600-85-C-0011		
6. AUTHOR(S) Anex, Robert P., Systems Control Technology, Inc.				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Systems Control Technology, Inc. Advanced Technology Division Palo Alto, CA 94303		8. PERFORMING ORGANIZATION REPORT NUMBER AEDC-TR-90-32, Volume II		
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Arnold Engineering Development Center/DO Air Force Systems Command Arnold AFB, TN 37389-5000		10. SPONSORING/MONITORING AGENCY REPORT NUMBER		
11. SUPPLEMENTARY NOTES Available in Defense Technical Information Center (DTIC).				
12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.		12b. DISTRIBUTION CODE		
13. ABSTRACT (Maximum 200 words) System identification is valuable for the estimation of gas turbine engine models because high fidelity models of jet engines are very useful for performance validation and because accurate models can aid in the understanding and prediction of jet engine phenomenon such as stall and surge. This report is an overview of techniques for system identification of gas turbine engine models. The emphasis in this report is on an overview of the set of techniques, solution of the <i>nonlinear</i> problem, and <i>practical</i> effectiveness rather than theoretical elegance. A detailed description of the operation of the SCIDNT parameter estimation code, which was developed by Systems Control Technology, Inc., is included. Although system identification is often regarded as a set of techniques for data processing, the overall technology has a broader scope which includes test planning, instrumentation specification, and choice of mathematical model structure, as well as the numerical methods of parameter estimation and the statistical techniques of interpreting results. These topics are all addressed herein.				
14. SUBJECT TERMS turbine engines stall system identification		stability surge post-stall		15. NUMBER OF PAGES 39
		gas turbine engines		16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT SAME AS REPORT	

PREFACE

This second volume of the final report is an overview of techniques developed for the system identification of jet turbine engine models. The emphasis of this volume is on the practical application of system identification using the SCIDNT parameter estimation code, which was developed by Systems Control Technology, Inc.

This volume provides an overview of the theory of parameter estimation, a description of how this theory is implemented in the SCIDNT computer code, and practical tips on the application of the SCIDNT code to the estimation of a general jet turbine engine model. The first five sections contain a description of parameter estimation theory and its implementation within SCIDNT. The last two sections offer specific guidelines for application of SCIDNT to a new engine model.

The first five sections of this report are not presented simply in an effort to be thorough or provide an archival reference, rather they are included because successful parameter estimation absolutely requires an understanding of the complete process. It is strongly recommended that potential users of these tools carefully review this entire volume prior to attempting estimation of an engine model.

The reproducibles used in this report were supplied by the authors.

Table of Contents

List of Figures.....	4
Nomenclature	5
1 Overview of System Identification.....	7
2 The Integrated System Identification Process.....	9
2.1 Iterative Nature of the Process.....	9
2.2 The Pretest Planning Phase	10
2.3 The Data Processing Phase.....	11
2.4 Model Validation.....	12
3 Solving Linear Least Square Problems.....	14
Geometric Interpretation	16
4 Parameter Estimation Performance Measures.....	17
4.1 Equation Error	17
4.2 Output Error.....	18
5 Iterative Optimization of the Performance Measures.....	20
5.1 Optimization Methods Used in Estimation.....	20
5.2 Calculation of Partial Derivatives of $\underline{y}(t_i)$ and of B.....	24
5.3 Mechanization in SCIDNT.....	24
6 Installing a New Model In SCIDNT	27
6.1 Introduction.....	27
6.2 The Model as a Subroutine of SCIDNT	27
6.3 Frequent Problems in Model Structure	28
7 Verifying and Debugging Model Installation.....	31
7.1 Tests to Verify Model Operation with SCIDNT.....	31
7.2 Interpreting SCIDNT Output for Debugging.....	32
7.3 General SCIDNT Output Interpretation.....	33
References.....	35

List of Figures

Figure 2.1	The Integrated System Identification Process.....	10
Figure 3.1	An Overdetermined Set of Equations.....	15
Figure 3.2	Geometric Interpretation of Linear Least Squares.....	16
Figure 5.1	Trust Radius Geometry	23
Figure 5.2	Parameter Optimization Flow Chart.....	26

Nomenclature

A	Matrix of independent variables in a regression
B	State estimation innovations covariance matrix
D	Controls into measurements distribution matrix
ϵ	Epsilon perturbation value (0.1%)
F	State dynamics matrix
f_d	Nonlinear state dynamics vector function
f_k	Nonlinear state kinematics vector function
g	Gradient of a performance measure
G_u	Control distribution matrix
G_w	Process noise distribution matrix
H	Hessian of a performance measure
H	States into measurements distribution matrix
K	Kalman gain matrix
L	Generic estimation performance measure
L	Luenberger observer gain matrix
L_e	Equation error estimation performance measure
L_f	Filter error estimation performance measure
L_m	Maximum likelihood estimation performance measure
L_o	Output error estimation performance measure
N_p	number of parameters
N_t	number of time points

N_u	number of controls
N_w	number of process noise sources
N_x	number of states
N_y	number of measurements
Q	Process noise covariance matrix
θ	Vector of unknown parameters
R	measurement noise covariance matrix
S	Diagonal matrix of singular values
t	time
U	An orthogonal matrix
u	control vector
V	An orthogonal matrix
w	process noise vector
ξ	Ridge regression parameter
x	state vector
Y	Matrix of dependent variables in a regression (usually a single column)
y	measurement vector
$(\hat{})$	Estimated quantity

Variable Types:

Bold Variables Vector quantities

Sans Serif Font Matrices

1 Overview of System Identification

System identification is a technology for determining a mathematical model of a dynamic system from observations of its response to its inputs. This technology has found application in a number of fields of engineering: vehicle dynamics¹, process control², electric power production and distribution³, aircraft aerodynamics⁴, medicine⁵, econometrics⁶, and structural dynamics⁷.

System identification is particularly useful for the estimation of gas turbine engine models. This is because high fidelity models of jet engines are very useful for performance validation and for control design and because accurate models can aid in the understanding and prediction of jet engine phenomenon such as stall and surge.

This set of notes is an overview of techniques for system identification of gas turbine engine models. These techniques have been developed over the last 10 years. The emphasis in these notes is on

- an overview of the set of techniques,
- solution of the nonlinear problem, and
- practical effectiveness rather than theoretical elegance (Many assertions will be stated without proof.).

Although system identification is often regarded as a set of techniques for data processing, the overall technology has a broader scope which includes test planning, instrumentation specification, and choice of mathematical model structure, as well as the numerical methods of parameter estimation and the statistical techniques of interpreting results. The overall scope of the problem can be illustrated with the gas turbine engine modeling example. In planning the effort, the following questions should first be addressed.

- What is the purpose of the system identification analysis? What parameters are known accurately *a priori* and which are not known? The ultimate purpose of the model will affect the choice of the structure of the model to be identified and the estimation accuracy required.
- What test inputs are acceptable from the point of view of safety of operation while the data are being collected? Small control excursions may not excite the dynamic modes of interest while large excursions may cause unsafe transients.

- **What aspects of the input and output of the system must be fetched from storage? Possible measurements on the engine include**
 - **control positions (e.g. stator angles, fuel flows, nozzle areas),**
 - **pressures,**
 - **temperatures,**
 - **flow rates.**
- **How accurately must these quantities be measured?**
- **What data length will give the highest probability of success? Test cell budget constraints may dictate short time records, yet short records may not yield accurate estimates.**
- **How high should the sampling rate be? A fast sampling rate may be needed in order to obtain information about parameters modeling comparatively fast engine dynamics such as stall behavior. But the processing hardware may not be able to store enough data at a high rate if the records are long enough to capture a slow acceleration.**
- **How complex must the mathematical model be? How many parameters need to be estimated? Do significant nonlinearities exist?**
- **Can the quality of the data be quickly assessed before more expensive processing algorithms are run?**
- **How can we assess the validity of the model determined by processing the test data?**

2 The Integrated System Identification Process

2.1 Iterative Nature of the Process

Figure 2.1 indicates an integrated identification procedure which applies to a wide variety of dynamic system types (e.g., structural dynamics, vehicle dynamics, etc.). This system identification process consists of an iterative loop of test planning, actual testing, and data processing. Two feedback loops can be vital to the overall success of the program. An inner loop is closed during the data collection phase. This inner loop checks the quality of the data produced by the tests. The checking is done in real-time or nearly in real-time. If the quality of the data is determined to be poor, then corrective action can be taken before possibly computationally expensive data processing begins. The test might need to be repeated with modified inputs. Poor data quality may be due to:

- failed sensor channels,
- excessively noisy sensor channels, or
- failure to excite dynamic modes of interest.

Data quality evaluation may be performed by a variety of real-time techniques including:

- pattern recognition methods that can extract features of the data channels (this mimics the human expert's visual inspection of data records as displayed by strip chart recorder or CRT.),
- fault detection filters⁸, or
- actual parameter estimation using algorithms configured for high computation speed⁹.

An outer loop of maneuver specification may also need to be closed about the entire identification process. The maneuver specification is a "boot-strap" process. A model of the system is needed in order to choose test input signals. The characteristics of the model which result from the identification data processing may indicate that additional input signals are required for complete model identification. If this is the case, then another set of data collection tests will be required.

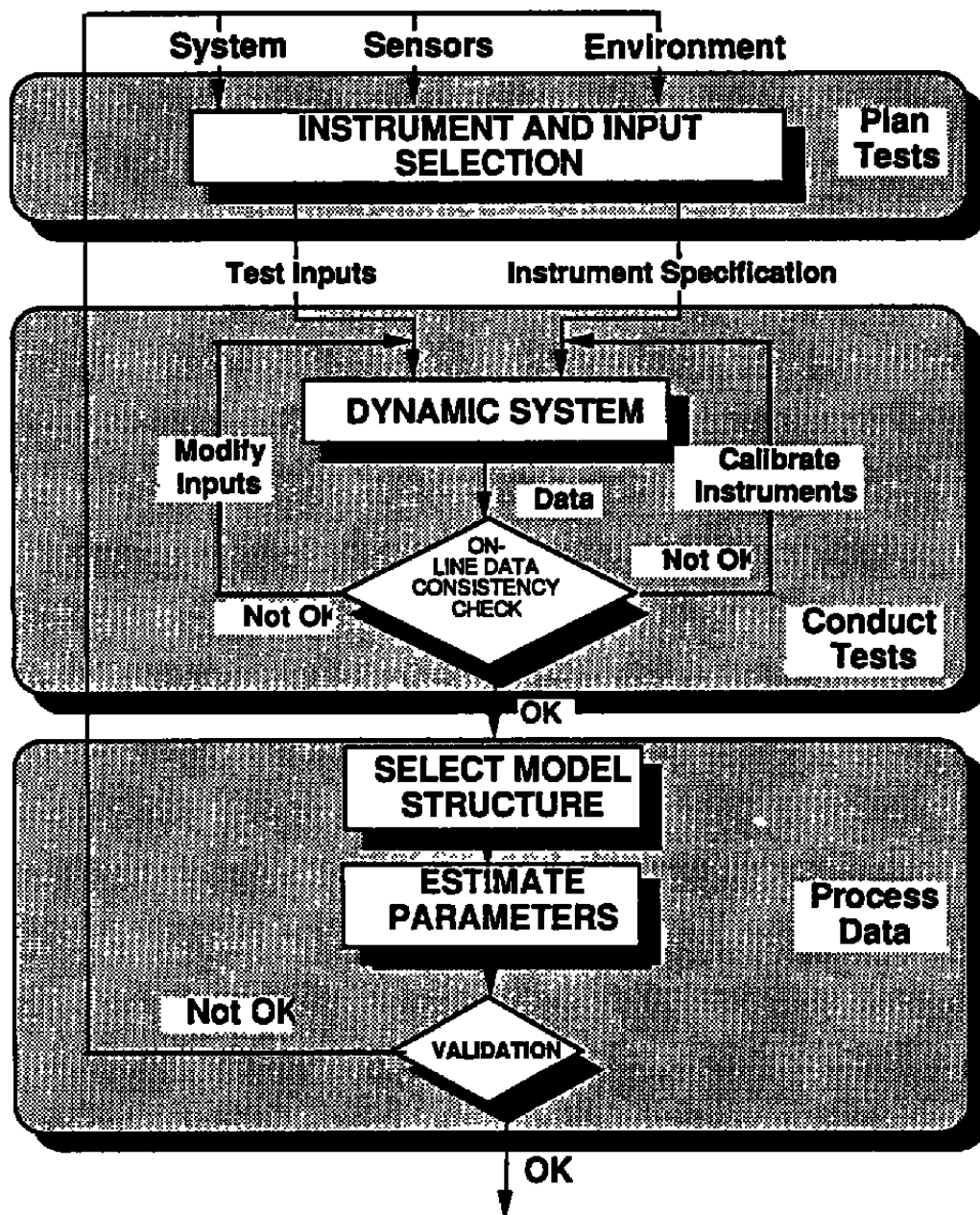


Figure 2.1 The Integrated System Identification Process

2.2 The Pretest Planning Phase

The pretest planning phase normally includes the specification of an instrument system and test inputs. The overall technology of system identification now extends to include analytic methods of specifying test input signals to minimize the uncertainty of estimates of parameters¹⁰ and of specifying sensor accuracy requirements¹¹.

2.3 The Data Processing Phase

The actual processing of the data requires three major steps. These are:

- model structure determination,
- parameter estimation, and
- model validation

The model structure determination phase¹² consists of processing the input/output data to determine the significant linear and nonlinear equations and associated parameters that are necessary to represent an observed system response. Questions addressed here include the determination of the order of the model (e.g., number of degrees of freedom) and a mathematical form (e.g., polynomial) to represent any nonlinear character in the dynamic equations. For linear dynamic systems, the determination of order is of primary importance. For nonlinear systems the determination of forms to represent nonlinearities has equal importance.

The estimation of unknown parameter values follows the determination of a suitable model structure. Numerical values of unknown parameters are determined by choosing them to optimize some performance index which measures how well the mathematical model represents the observed data. Possible performance criteria include:

- minimization of sum of squared fit errors, and
- minimization of the autocorrelation of the fit errors.

Fit error is the difference between the observed response of the dynamic system and the simulated response of the system model to the observed inputs.

The determination of model structures and the estimation of parameter values are often done in parallel. Model structures are determined by fitting several competing candidate models (specified by the user) to the observed system response. The model structure which gives the "best" fit of the data is the structure chosen.

Several criteria may be used to evaluate the closeness of the fit. Simple mean square fit error is not a suitable criterion for the comparison of all candidate models. The use of this criterion will always lead to the choice of highly complex models, since adding degrees of freedom to the model always leads to reduced mean square fit error. A more useful criteria, when evaluating candidate models having different numbers of parameters are the mean

square prediction error¹³. These criteria weight fit error against the number of free parameters in the model.

We have found that a two-stage process is effective for determining the structure of the model and estimates of the parameters. First, candidate models are evaluated by choosing their parameters to minimize fit error or mean square prediction error. This evaluation is done using a numerical scheme which accurately evaluates the performance index but which may not accurately evaluate the parameter estimates themselves. Once the model structure is established in this way, the parameter estimates are refined using a scheme which gives more accurate parameter estimates. For dynamic systems, a computationally efficient method which is effective for model structure determination is the equation error minimization method. Parameter estimates may subsequently be refined using output error minimization or combined state and parameter estimation methods.

2.4 Model Validation

A good criterion for the validation of a model is the use of the model to predict new data. A typical procedure might be to use, say, 80% of the available data to determine the model structure and parameter values. Then the resulting model would be used to predict the remaining 20% of the data. The degree of validation achieved can then be interpreted from the accuracy of the prediction.

Statistical performance criteria include:

- mean square fit error,
- mean square prediction error, and
- whiteness (statistical independence) of residuals.

Whiteness can often be evaluated effectively by visual inspection of plots of the observed data superimposed on plots of the predicted data. Plots of the residuals themselves may also be used.

- RMS residual: The root-mean-square of the difference between recorded data and the predictions of that data using the identified model. This is probably the most popular purely statistical performance index.
- Autocorrelation of residual: This is a measure of the whiteness of the error between recorded data and the predictions of that data.
- Correlation of input with residual.

- Prediction error
- Parameter covariance.

Finally, validation should include comparison of the model determined from system identification with models available *a priori*.

3 Solving Linear Least Square Problems

In order to apply the parameter estimation to a problem, one must understand the basic goals and principles involved. Therefore, in this and the following chapter a brief overview of parameter estimation theory is presented. There is slightly more detail here than required for the reader interested only in application, but it provides an archival reference directly relevant to gas turbine engine model estimation.

We begin the study of parameter estimation with the linear least square problem. This is more precisely called the estimation of parameters in linear static models using a least sum of squared errors criterion. We begin with this method for several reasons.

- It is very easy to solve. We will see that there are simple algorithms for calculating the best parameters.
- The study of this method introduces many of the ideas that will be used in other, more advanced methods.
- The method is useful in itself for estimation.
- The more advanced methods such as filter error and output error are fundamentally nonlinear, even when working with linear models. However, each of them operates by approximating the nonlinear estimation problem as a series of linear problems.

In order for linear least square fitting to work, we must have a model in the form:

$$y_i \equiv \sum_{j=1}^m a_{ij} \theta_j + v_i$$

Where y is called the "dependent variable". The terms a_{ij} are called "independent variables". The term v_i represents all of the things that affect the value of the dependent variable but that are not represented by the independent variables. v_i is sometimes called "the noise". As we will see later, it is traditional to make various statistical assumptions about the characteristics of v_i . The more simple assumptions allow us to make statements about the statistical characteristics of the parameter estimates and about the ability of the estimated model to predict the behavior of the real world. Since these more simple statistical assumptions about v_i are usually wrong, the statistical conclusions about results are usually wrong also.

The above model can be written in matrix equation form as:

$$y = A \cdot \theta + v$$

where v is an N_t column vector of measurement noise,

A is a $N_t \times N_\theta$ rectangular matrix of measured independent variables, and

θ is an N_θ column of model parameters.

This equation will appear as represented in Figure 3.1. Note that this is usually an overdetermined set of equations, meaning that there are more (usually many more) equations than unknowns.

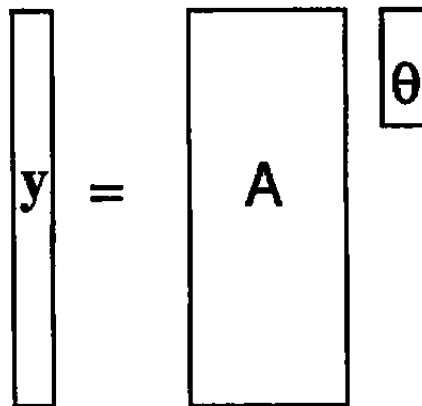


Figure 3.1 An Overdetermined Set of Equations

A vector of the errors between the model outputs, $A\theta$, and the measured outputs, y , is defined as r , and referred to as the residual vector. When the residual vector is smallest it is perpendicular to the column space of A . This is illustrated in Figure 3.2. Where a linear model is defined by the vector labelled θ . For any value of θ , certain values of the outputs y_1 and y_2 result. Obviously the shortest residual vector (and therefore least error) results when the residual vector is perpendicular to the θ vector. If the residual vector is perpendicular to the column space of A , the following must be true:

$$A' \cdot r = A' \cdot (y - A \cdot \theta) = 0$$

$$A' \cdot y = A' \cdot A \cdot \theta$$

$$\theta_{ls} = (A' \cdot A)^{-1} \cdot A' \cdot y$$

Thus geometry has led us directly to the fundamental equation of least squares theory and a definition of the least square error estimate.

Geometric Interpretation

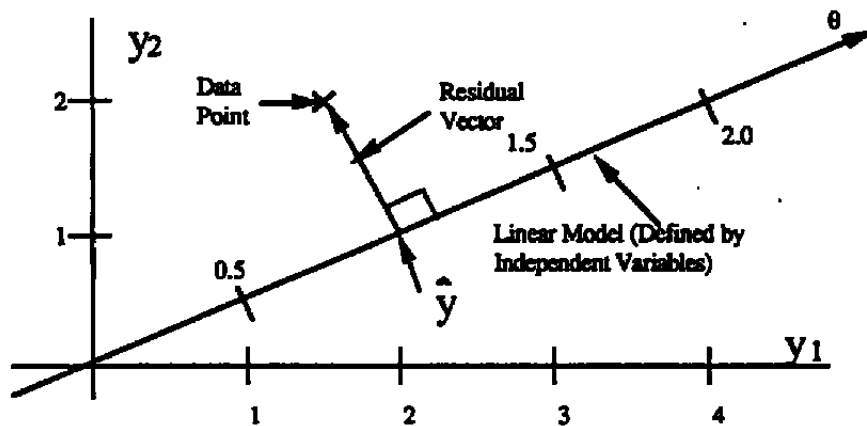


Figure 3.2 Geometric Interpretation of Linear Least Squares

4 Parameter Estimation Performance Measures

A large number of methods exist for performing system identification data processing. The best algorithm for any given application depends strongly on the type of model and on the nature of the available data. No one type of processing algorithm can handle all possible applications.

This section outlines two processing methods which have been found to be effective in a variety of applications. These methods are:

- equation error minimization methods,
- output error minimization methods.

The SCIDENT estimation code is capable of performing both of these types of parameter identification. Generally gas turbine engine model estimation is performed using output error minimization methods.

4.1 Equation Error

The equation error minimization methods estimate unknown parameters by choosing them to minimize a performance index. A continuous dynamic system must be represented as:

$$\dot{x} = f(x, u, t, \theta + w) \quad (4.1)$$

where θ is a set of p unknown parameters and w is a time-varying unobservable disturbance. An analogous formulation exists for a discrete dynamic system. The performance index $L_e(\theta)$ to be minimized is:

$$L_e(\theta) = \sum_{i=1}^m (\dot{x}(t_i) - f[x(t_i), u(t_i), t_i, \theta])^2 \quad (4.2)$$

The equation error minimization method is often called the least squares method because of the form of the performance index $L_e(\theta)$.

The effective use of the equation error minimization requires the a priori determination of system states x , controls u , and state derivatives \dot{x} over the time interval of the test. A priori here means that these quantities must be determined before the unknown system parameters are estimated. This determination may be done using direct measurements or using system

characteristics that are independent of the parameters. For example, an unmeasured state derivative may be determined by (very carefully) numerically differentiating a measured state time history.

The term w is a stochastic quantity which represents unmeasurable process disturbances in the system. For a gas turbine engine, w consists of disturbances such as inlet distortion and unmodeled, high-order aero-thermo effects.

The special advantage of the equation error minimization method lies in the fact that many nonlinear dynamic system functions $f(x, u, t, \theta)$ are linear in the parameters θ . In other words,

$$f_k [x(t_i), u(t_i), t_i, \theta] = \sum_{j=1}^p \{ \theta_{k,j} f_{k,j} [x(t_i), u(t_i), t_i] \}$$

The functions $f_{k,j}$, $j=1,2,\dots,p$ are independent of the p unknown parameters θ_j . The parameter values that minimize $L_\theta(\theta)$ can be found explicitly using linear algebraic operations¹⁴. The disadvantages of the equation error minimization method arise primarily from the requirement for very accurate measurements of states and controls. States will inevitably be measured with some error. No measurement at all may be available for other states.

4.2 Output Error

Output error minimization methods, like equation error minimization methods, estimate unknown parameters by choosing them to minimize a performance index. The dynamic system must be represented as

$$\dot{x} = f(x, u, t, \theta + w, \quad (4.4)$$

$$y = h(x, u, t, \theta) + v \quad (4.5)$$

where θ is a set of p unknown parameters and v is a time-varying, unobservable, additive measurement error. The performance index $L_o(\theta)$ to be minimized is

$$L_o(\theta) = \sum_{i=1}^{i=m} \frac{\{y_k(t_i) - \hat{y}_k(\theta, t_i)\}^2}{w_k^2}$$

Here $y_k(t_i)$ is the k^{th} channel of measured system output at time t_i . \hat{y}_k is the k^{th} channel of system output y predicted for time t_i by solving the

system state equations and measurement equations using the measured system inputs $u(t_i)$ and the a priori parameter values θ .

The effective use of the output error minimization requires the very accurate measurement of system inputs u and the measurement of system outputs y . The method will tolerate errors in the measurement of y .

The term v is a stochastic quantity which represents instrument measurement errors, e.g., analog-to-digital quantization noise.

The special advantage of the output error minimization method, with respect to the equation error method, is that the measurement requirements are greatly relaxed. The method does not require the accurate measurement of all state and state derivatives. Rather, it is effective using noisy measurements of the limited number of outputs that are available.

The actual determination of the parameter values θ that minimize the performance index $L_o(\theta)$ is computationally more complex than the minimization $L_e(\theta)$. This is because $L_o(\theta)$ is a nonlinear function of the parameter set θ . Finding the minimizing parameter set requires an iterative numerical scheme^{15,16}. The application of such numerical methods is often not straightforward.

The principal disadvantage of the output error minimization scheme is that it does not explicitly allow for the presence of unmodeled disturbances in the state dynamics. Such disturbances are represented in the equation error method by the term w . "Process noise" is the term often used to describe these unmodeled effects.

It should be noted that the output error method can account for system dynamics disturbances of unknown magnitude if the form of these disturbances is accurately represented. The disturbance w must be explicitly represented as

$$w = w(t, \theta) \quad (4.7)$$

The unknown elements of the disturbance are represented using part of the unknown parameter vector θ .

5 Iterative Optimization of the Performance Measures

The following paragraphs contain a discussion on the optimization methods used in estimation, modification for dynamic system likelihood function, and calculation of partial derivatives of $y(n)$ and B .

5.1 Optimization Methods Used In Estimation

Finding maximum likelihood parameter estimates requires the use of an iterative optimization method to maximize the likelihood function with respect to the unknown parameters. This may be done using a Newton-like optimization method that minimizes the likelihood function. A Newton method makes successive approximations to the minimum or maximum of a nonlinear function of several variables $\underline{\theta}$ using

$$\underline{\theta} \leftarrow \underline{\theta} + \Delta \underline{\theta} \quad (5.1)$$

where the change in the parameter vector $\Delta \underline{\theta}$ is the solution of

$$\underline{H} \cdot \Delta \underline{\theta} + \underline{g} = \underline{0} \quad (5.2)$$

\underline{H} and \underline{g} are the Hessian and the gradient of the nonlinear function being minimized.

In many estimation problems (not necessarily those for linear dynamic systems), the likelihood function is a sum of squared nonlinear functions.

$$L = \sum_{i=1}^{i=Nt} \{v^2(i)\} \quad (5.3)$$

These functions $y(i)$ are the differences between actual observations and observations estimated by some type of model. The fact that the function has the special structure (sum of squared terms) makes the application of Newton methods particularly efficient. It is possible to approximate the Hessian knowing only the first derivative of each of the nonlinear functions with respect to the unknown parameters. This is the basis of the Gauss-Newton (GN) algorithm.

$$\underline{H} \approx \underline{J}' \cdot \underline{J} \quad (5.4)$$

where

$$J_{i,j} = \frac{\partial v(i)}{\partial \theta_j} \quad (5.5)$$

The rectangular matrix J is called the Jacobian of the estimation problem. The exact expression for the gradient is just

$$\underline{g} = \sum_{n=1}^{n=N_t} [J_n v(n)] \quad (5.6)$$

Execution of each iteration requires the evaluation only of $v(n)$ and of J . The Levenberg-Marquardt (LM) algorithm^{17,18} is a modification of the GN algorithm. This modification aids convergence when working on highly nonlinear or ill-conditioned problems. LM approximates the Hessian as

$$H \cong J^T J + \zeta I \quad (5.7)$$

where I is the identity matrix of appropriate dimension and ζ is a positive scalar called the Marquardt parameter. If ζ is very small, the LM algorithm behaves just like the GN algorithm. If ζ is very large, LM behaves like a gradient search. The trade-off is that on problems of moderate difficulty, GN converges much more quickly than a purely gradient scheme. However, the gradient scheme is more robust and will perform more reliably on highly nonlinear or ill-conditioned problems. The complete LM algorithm adjusts ζ during the course of iterative approximation to the minimum. The algorithm starts with some nominal value of ζ . This starting value may be zero. If an iterative step fails to make an improvement in the cost function, LM increases ζ and tries the step again. If an iterative step successfully decreases the cost function on its first try, LM decreases ζ . The usual factors for changing ζ are 2 and 1/2.

SCIDNT chooses the starting value of ζ automatically to limit the trial step size of the first iteration. The user can choose the magnitude of this initial limit using the program control parameter RADIUS. The default value for RADIUS is 1.0. Choosing the initial RADIUS value as 1.0 means that the first normalized trial step length will satisfy

$$\frac{|\Delta \theta_{\text{trial}}|}{|\theta|} \leq 1.0 \quad (5.8)$$

(See Figure 5.1.) Scaling the elements of the gradient and Hessian may improve the conditioning of the linear system that must be solved for $\Delta\theta$. A choice for a set of scale factors that is sometimes effective is the magnitude of the parameters. The scaled Hessian becomes H_S and the scaled gradient becomes g_S where

$$H_S = S \cdot H \cdot S \quad (5.9)$$

$$g_S = S \cdot g \quad (5.10)$$

The square scaling matrix S is

$$S = \begin{bmatrix} \theta_1^{-2} & 0 & 0 & 0 \\ 0 & \theta_2^{-2} & 0 & 0 \\ 0 & 0 & \ddots & \vdots \\ 0 & 0 & \dots & \theta_{N_{th}}^{-2} \end{bmatrix} \quad (5.11)$$

The Levenberg-Marquardt approximation is then applied to H_S :

$$H \equiv H_S + \zeta \cdot I. \quad (5.12)$$

SCIDNT applies this scaling rule automatically. Normally, scaling is not a concern since it is done inside the estimation algorithm and does not appear on any of the SCIDNT outputs. The only place where it does appear is in the Hessian and Hessian eigensystem listing at the end of the summary printout (SCIDNT.OUT).

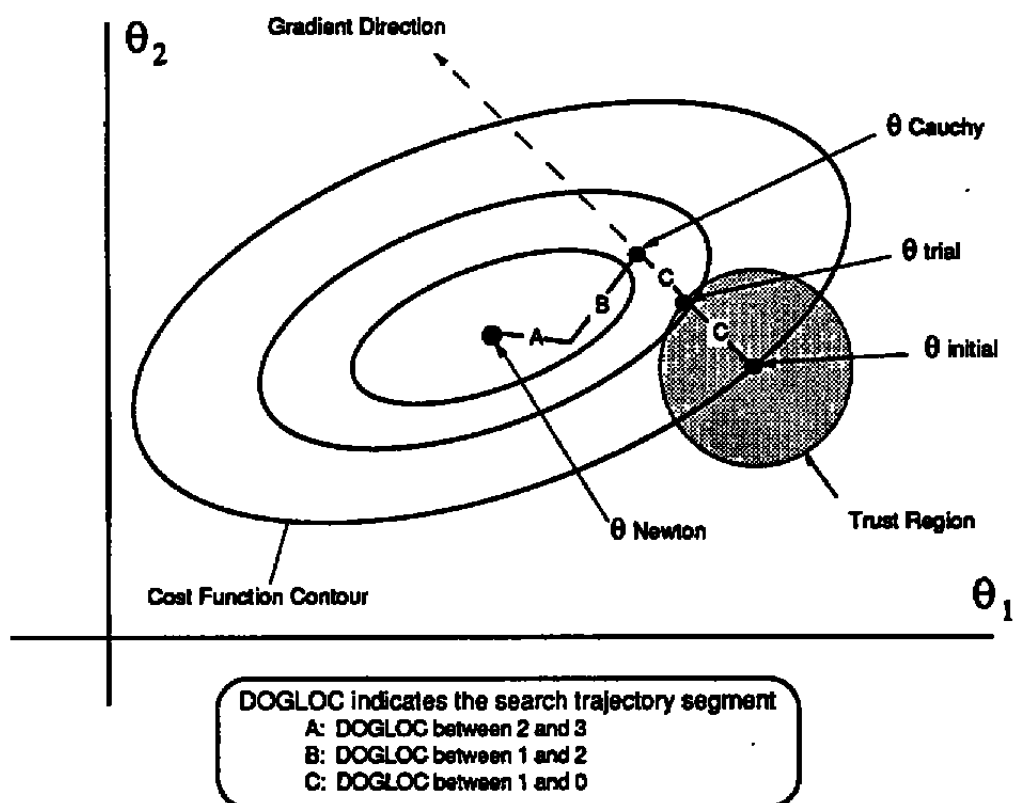


Figure 5.1 Trust Radius Geometry

5.2 Calculation of Partial Derivatives of $y(t_i)$ and of B

The modified LM algorithm described above requires the evaluation of the negative log likelihood function (L), innovations (y), innovations covariance (B), and the partial derivatives of y and of B with respect to each of the unknown parameters. The previous section showed how to evaluate the L , $y(t_i)$, and B . It remains to evaluate the required partial derivatives. Several publications discuss analytic formulations for these partial derivatives¹⁹. The partial derivatives themselves satisfy certain linear, constant coefficient differential equations. These may be solved in parallel with the system model equations in order to evaluate the partial derivatives. A more effective method often is to approximate the needed partial derivatives using finite differences. This approach has been studied extensively for the LM algorithm²⁰. An algorithm that uses these approximations is called a "derivative free analog" of the exact LM method. SCIDNT computes estimates of partial derivatives by

$$\frac{\partial y(t_i)}{\partial \theta_j} \equiv \frac{\Delta y(t_i)}{\Delta \theta_j} = \frac{y(t_i)|_{\theta+\Delta\theta} - y(t_i)|_{\theta}}{\Delta \theta_j} \quad (5.13)$$

The partial derivative of the innovations covariance may be estimated in a similar fashion. To approximate partial derivatives in this way, the model producing the output sequence must be run once for the nominal value of parameters and once for each perturbed parameter.

The need to repeatedly run the model for each perturbed parameter has a direct impact on the required model structure. A model which is to be used in the SCIDNT estimation code must not have any of non-reset counters, flags, and lags which are common in models designed to be run only once. The model outputs must vary between calls only due to the perturbed estimation parameters and any "model memory" from previous propagations will carry over into the next and invalidate the partial derivative calculations.

5.3 Mechanization in SCIDNT

The mechanization of the search algorithm in SCIDNT uses the following steps (Figure 5.2): There are three major loops in the algorithm: the iteration loop, the measurement sensitivity calculation loop, and the trust radius search loop. Each iteration loop produces a new estimate of the parameters

optimizing the cost function. Each cycle through the measurement sensitivity calculation loop produces a set of sensitivities of the measurements to one parameter. Each cycle of the iteration loop requires N cycles through the sensitivity calculation loop. Each cycle of the iteration loop also requires one or more cycles of the trust radius search loop. Given the Newton step $\Delta\theta_N$ computed by the iteration loop, the trust radius search loop tries to find a trial parameter set θ_{trial} that actually does improve the cost function.

To start SCIDNT, an initial guess for parameters θ and the initial search radius must be made. The iteration loop then computes an initial evaluation of the cost function $L(\theta)$. Next the measurement sensitivity calculation loop computes the sensitivity of each measurement to each parameter. It does this using the parameter perturbation approximation method described in equation 5.13. SCIDNT perturbs each parameter one at a time by ϵ and then runs the filter with the perturbed parameter value. It then approximates the measurement sensitivities using the differences between the nominal and the perturbed parameter measurement evaluations. SCIDNT next computes the gradient and Hessian. Having the gradient available allows computation of convergence criteria.

SCIDNT next enters the trust radius search loop. This loop (shown in simplified form in Figure 5.1) ensures that each iteration actually improves the cost function $L(\theta)$. The trust radius search loop limits the size of the $\Delta\theta$ step. The idea here is that for a long step, the GN approximation to the shape of the cost function may be inaccurate and may actually lead to a worse cost function than the current θ value. Even if the long step is a failure however, there should exist some shorter step that does improve $L(\theta)$. The trust radius search first finds a trial set of parameters θ_{trial} that, based on the GN approximation, should minimize the cost function within the constraint $|\theta_{trial}| < \text{Radius}$. If the step is successful [i. e. $L(\theta_{trial}) < L(\theta)$], the search will increase Radius to try a still larger step. If the step is not successful, the search will decrease Radius and try again.

The trial steps that SCIDNT will take are illustrated in Figure 5.1 by the lines labelled A, B and C. The C segment is in the gradient direction from the initial θ values. The A segment is in the Newton direction, based on the approximation of the Hessian described above. The B segment is a "dogleg" connecting the gradient and Newton directions. The location of the B segment is set by the user through the *bias* input. If the system is fairly linear, the Hessian approximation should be accurate and a step in the Newton direction should be successful. If this is not the case, a shorter Newton step or a step in the gradient direction may still improve the cost function. Since the computed gradient will be accurate at least in the immediate region around the initial θ values, a very short step in the gradient direction must improve

the cost function. Thus the path A-B-C gives provides a logical blending of the robust gradient optimization with the Newton optimization with its superior convergence properties. The SCIDNT output *DOGLOC* indicates the search trajectory segment used for a given step as indicated in Figure 5.1.

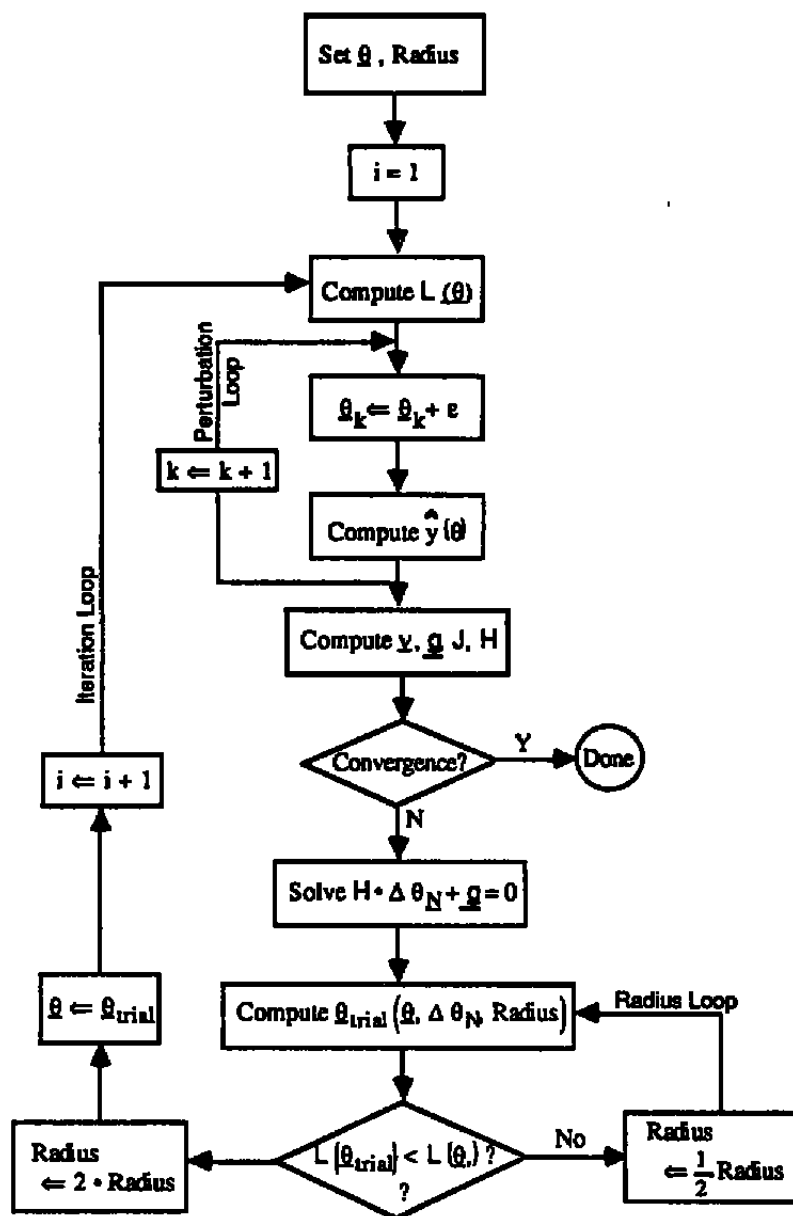


Figure 5.2 Parameter Optimization Flow Chart

6 Installing a New Model In SCIDNT

6.1 Introduction

This chapter discusses the installation of a new model into the SCIDNT parameter identification code. It is assumed that an output error estimation procedure will be employed and that the model is embodied in a stand-alone simulation. This is usually the case for gas turbine engine model estimation. This chapter is not intended to cover all possible modifications of SCIDNT for different parameter estimation applications.

The preceding chapters of this volume were presented to explain to the novice user what the SCIDNT code does and why. This was done because it is far better to work with SCIDNT as an informed user, than by trying to follow some apparently arbitrary instructions. It is strongly recommended that before attempting to install a new model in SCIDNT, the user read the first five chapters of this report with the goal of gaining an understanding when and with what purpose the SCIDNT code will run the model. Thus the user may determine if the model is installed so as to correctly serve these purposes.

6.2 The Model as a Subroutine of SCIDNT

Engine models are installed in the SCIDNT estimation code as a set of subroutines called from a single module. Thus SCIDNT makes a single subroutine call (to DYNMOD in the current version) to run the model. It is assumed here that a model to be installed in SCIDNT is currently in the form of a stand-alone computer simulation. The top level subroutine which connects the engine model to SCIDNT can usually be a modified version of the original MAIN module of the stand-alone simulation.

The argument list of the model module is not invariant. Each model to be estimated has different characteristics and generally require slightly different information from the estimation code. A typical list which embodies all of the required values is:

Subroutine Dynmod(Tout, Delt, U, NU, NY, Ifirst, Ibomb, Y, P).

Tout is a scalar variable which tells the simulation at what time SCIDNT next expects to receive model outputs. *Delt* is the delta time between the value of *Tout* at the last call and the current call. *U* is a vector of inputs as recorded during testing which must drive the model (such as nozzle throat area and fuel flow). *NU* and *NY* are integers which define the sizes of the *U* and *Y* vectors respectively. *Ifirst* is a logical which indicates if this is the first call to the simulation. This allows certain model initialization to be done only once. *Ibomb* is a logical which the simulation returns to SCIDNT to inform it that

the simulation has failed in some way. Y is a vector of model outputs which the simulation returns to SCIDNT. P is a vector of the model parameters which are being estimated. These parameters are provided by SCIDNT and must be used in the model routines as model variables.

The simulation must operate such that SCIDNT passes it the value $Tout$, and the simulation integrates forward in time to this value and then returns to SCIDNT the values of the outputs, Y , at that time. Thus checks must be inserted into the top level module of the simulation to insure that it will return at this time. It is possible to call an interpolation routine (one is included in the SCIDNT code) if needed to produce the outputs precisely at $Tout$. This may be necessary if $Delt$ is not an even multiple of the simulation integration time step.

The user should remember what it is that SCIDNT is doing and insure that the installed model accommodates these requirements. The simulation must be modified to integrate forward to $Tout$ and return the model outputs, Y , at that time. The model must use as inputs those recorded during test and provided by SCIDNT in the vector U . The model must use the parameter values provided in the vector P . Obviously, if these parameters are to be estimated they must affect the model outputs. If they do not, SCIDNT will give the user a message that these parameters are unidentifiable. Model parameters which are not to be estimated, but which the user wishes to change easily, can also be passed in through this vector and need not affect the model outputs.

The values of U and Y which were measured during test must be made available to SCIDNT by locating them in a file. The default file type is double precision binary, but any formatted file type is possible if the user is willing to edit SCIDNT to read that file type. The data is read in by the routine INREAD. The SCIDNT code structure and detailed setup and operating instructions are located in the *SCIDNT User's Guide*.

6.3 Frequent Problems in Model Structure

There are number of common simulation characteristics which will cause the user difficulty when trying to install a simulation containing them in SCIDNT. A few of these will bear enumeration here as examples to the user of common pitfalls.

Generally, any model characteristic which has "memory" will cause problems. Since SCIDNT will make repeated calls to the simulation, these "memories" will contain erroneous values at the start of all but the first propagation. Some common sources of "memory" are counters and flags, state variables, and approximations to lags which are not reset. This type of model structure is not a problem in the original simulation which would

only propagate once per run, but create large errors as SCIDNT repeatedly calls the simulation.

If the engine simulation uses a variable time step integration method special care must be taken to return to SCIDNT at the correct time. If a variable time step is used, a check on the time and a possible interpolation of the outputs will suffice. If a predictor/corrector integration method is used, extra care must be taken to insure that a return is not made on a tentative predictor pass rather than at the end of an actual integration step.

If a simulation uses inputs which are called from look-up tables or is a closed-loop simulation, special care must be taken to insure that the model is driven by the measured inputs. It is very important for output error estimation methods that an accurate measurement of the test inputs be used to force the model. Particular attention should be paid to insuring that simulation read and look-up statements do not over-write the measured inputs from SCIDNT.

Similarly, model parameters are often functions of operating conditions and subject to various resets based on engine operation. If model parameters are to be estimated, their values must be received from SCIDNT through the P vector, and not modified within the simulation. The user should search the code for any occurrences of estimation parameters on the left side of an equal sign and the use of the parameter name in look-up and equivalence statements.

Another consideration is that the simulation must represent as nearly as possible the exact article which was tested. Care should be taken to insure that important components such as inlet plenums are included in the simulation. In addition, the measurements must correspond to the model outputs, Y. If model outputs do not exactly correspond to the measured data, the model should be modified to produce the correct outputs.

Users should also scan the SCIDNT code for any COMMON names and subroutine names which may be present in both SCIDNT and the engine simulation. Obviously, these will cause difficulty in combining the two bodies of code.

There are both double and single precision versions of SCIDNT in release at this time. AEDC and WRDC have received double precision versions, and it is suggested that these are used as they are more accurate than the single precision versions. If the engine simulation to be identified is not double precision, it should be converted or care must be taken to insure that it will interface properly with the SCIDNT code. Precision mismatch is a very common error in code which attempts to use different precision variables. This is very easy error to make and very difficult to de-bug since memory is overwritten in apparently random times and places.

Obviously, the SCIDNT code must be linked with all modules including the engine simulation and all of its INCLUDE and COMMON files. Care must be taken during the linking process to insure that the I/O units used by SCIDNT are not used in the engine simulation also. Most engine simulation output statements can be disabled since SCIDNT will provide output. The I/O units used by the SCIDNT code are detailed in the *SCIDNT User's Guide*.

7 Verifying and Debugging Model Installation

7.1 Tests to Verify Model Operation with SCIDNT

Having created an executable image of SCIDNT with an engine simulation the user should make a series of organized SCIDNT runs to insure proper operation of the model within SCIDNT, as well as the proper interaction of SCIDNT with the model.

The first SCIDNT debug run should be a simple propagation of the model. Set the SCIDNT input parameter LIDENT equal to .FALSE. for this run. This will run the model once through, and not make the repeated model calls required to compute the Jacobian, Hessian, or estimate model parameters. This run should precisely match the operation of the simulation as a stand-alone model. (A run of the stand-alone model using the same inputs and configuration should be made to use as a test case.)

Having passed this first test, the simulation should then be tested for "memory". The user should specify LIDENT equals .TRUE. and put three or four parameters numbers into the IDENT vector in the SCIDNT input. One of the parameters in IDENT should be a parameter which is known to have no effect in the model outputs. This may be a dummy parameter which is not present in the model at all. Also, the user should put a zero in the PERT vector corresponding to one of the parameters of IDENT other than the one which has no effect in the outputs. This run should return the information that the "no effect" parameter and that corresponding to the zero in Pert are "nonidentifiable". If this is not the case, the model outputs were different for model propagations where these parameters were perturbed. This should not happen, since in one case the parameter should have no effect in the outputs and in the other the parameter was perturbed by a zero amount (not actually changed at all). If in fact the model outputs did change, it indicates that the model is non-repeatable, probably due to some "memory", and the user must find the problem and rectify it.

If SCIDNT has passed the above test runs, a simple estimation of one parameter of "known" value is recommended. A parameter such as the heat value of fuel is a good choice. This single parameter which has a large and obvious effect on the model will allow the user to detect obvious flaws in the estimation procedure and tune the user specified inputs to SCIDNT for his particular problem. This run will generally give the user some experience with estimating his or her model and gain confidence in the code installation. (Be sure to eliminate or reset the PERT vector prior to this run.)

7.2 Interpreting SCIDNT Output for Debugging

The SCIDNT code outputs a complete history of the estimation process. If the user has an understanding of the estimation procedure as described in the first five sections of this volume, this output should be quite intuitive. Although a description of the SCIDNT output is included in the *SCIDNT User's Guide*, and a few examples of how to interpret this output to detect common problems will be presented here.

The SCIDNT output is divided into two sections. The first echoes back the user inputs and default values which will be used during the estimation run. This information includes such things as parameters to be estimated, measurements used, measurement weightings, and number of time points. The second section describes the optimization procedure and summarizes the results at the end of the run. If the user is not familiar with the optimization procedure used by SCIDNT, he or she should review Section 5 before proceeding further.

One common problem when estimating gas turbine engine models is that the Newton step will be inaccurate due to the highly nonlinear nature of the problem. If the initial trust radius is large the first step which SCIDNT takes may cause the simulation to fail. Failure is usually due to the inability to balance terms due to a very large parameter change. This problem will be obvious due to the failure and will be reflected in the output by a large value in the LENGTH OF STEP portion of the output. The remedy is to reduce the RADIUS input to SCIDNT.

A similar problem will occur if the GROW input is set too large. This is the factor by which the RADIUS can be enlarged after a successful step. (If this doesn't sound familiar, review Section 5.) If GROW is too large, a very large step will be taken after one successful step and this will cause the simulation to fail. The user must evaluate what range of values are reasonable for the parameters to be estimated and set the RADIUS and GROW inputs accordingly.

Co-linearity of two or more parameters is another problem which may be encountered during optimization. This occurs when two model parameters have the same effect on the model outputs. This does not mean that they have the same role in the model, but that they are not identifiable from the chosen set of outputs. This can be detected by looking at the eigenvalues and eigenvectors of the Hessian which are printed at the end of each run. If two parameters are co-linear, they will have similar eigenvectors.

For estimation problems containing a large number of parameters this is difficult to determine. An alternative is to find the null-space of the Hessian itself. A convenient way to do this is using the Singular Value Decomposition of the Hessian, which produces a basis for the null-space

directly. This basis will show which rows and column of the Hessian are linearly dependent, and therefore which parameters are co-linear. For more detail on the basic sub-spaces of linear equations and singular value decompositions, see any linear algebra textbook. Note that this type of analysis may be required if SCIDNT returns the information that the Hessian is non-invertible. SCIDNT will only return this message if there are numeric problems such as those caused by co-linearity. If individual parameters are nonidentifiable, SCIDNT will remove them from the estimation and inform the user that they are not to be estimated.

7.3 General SCIDNT Output Interpretation

The SCIDNT output details the optimization procedure described in Section 5.3. This output and how it corresponds to the optimization process will be briefly described for reference.

The first thing that SCIDNT does during an identification run is repeatedly call the model to form an estimate of the gradient and Hessian at the nominal point. Recall that these are the first and second partials of the cost function with respect to the estimation parameters, and as such are linear in nature and only apply to one set of parameter values (one point in parameter space).

Once the gradient and Hessian have been computed the optimization procedure is begun. Optimization of the cost function using the gradient and Hessian approximation at one point is called an iteration. Thus each iteration is made up of an optimization which in turn is composed of a set of optimization steps. The entire estimation is made up of a series of iterations.

At the start of each iteration SCIDNT reports the current cost function value, the value of the MAXCOS convergence parameter, and the values for a Newton step. A Newton step may never be taken, but this information allows the user to evaluate the accuracy of the Hessian approximation. The Newton step information is composed of the value of each parameter, its standard deviation estimate, the gradient of the cost function with respect to it, and the normalized step which would be taken. SCIDNT also reports the norm length of the Newton step, and the predicted change in the cost function for the Newton step.

SCIDNT then begins the optimization with a step the length of which is determined by the RADIUS value. The optimization then proceeds as described in Section 5, with the result of each trial step being reported by SCIDNT. The length of the step and the change in the cost function value are output. If an improvement occurs the step will be accepted and a new iteration will be begun. This means a new gradient and a new Hessian will be computed, and another optimization begun.

For each step in an optimization the value of DOGLOC is echoed to tell the user if the step taken represents a step in the Newton, gradient, or dogleg direction. (See Section 5.3 for a description of the optimization procedure and these terms.) Similarly the value of MAXCOS gives a measure of the degree of convergence achieved. MAXCOS is the maximum of the cosines of the angles between the column space of the Jacobian transpose and the residuals. At a singular point, these vectors will all be perpendicular, thus the cosines will all be zero. Therefore as the estimation nears convergence, the maximum cosine (MAXCOS) will approach zero. MAXCOS provides a measure of convergence which is always between 1 and 0. (MAXCOS = 0.05 is a good value for convergence in most cases.)

When convergence is achieved, or the maximum number of iterations or steps (as set by the user) is reached, SCIDNT will stop and give a summary of the estimation. If the code stops due to convergence, the user will be informed of this and the convergence parameter limit which was reached will be reported. If the run ends due to an iteration or step limit, the user must realize this by lack of any other message and the knowledge that the maximum number of one of these two items was reached. Upon completion of an estimation run SCIDNT will report the final parameter values, the value of the cost function and the eigen analysis of the Hessian.

References

- 1 Hall, Jr., W.E., "System Identification-An Overview," Naval Research Reviews, Vol. 30, No. 4 April 1977, pp 1-20.
- 2 Gustavsson, I., "Survey of Applications of Identification in Chemical and Physical Processes," Proceedings of the 3rd IFAC Symposium, The Hague/Delft, The Netherlands, 12-15 June, 1973, pp 67-85.
- 3 Baeyens, R., Jacquet, B., "Applications of Identification Methods in Power Generation and Distribution," Proceedings of the 3rd IFAC Symposium, The Hague/Delft, The Netherlands, 12-15 June 1973, pp 1107-1121.
- 4 Trankle, T.L., Vincent, J.H., and Franklin, S.N., "System Identification of Nonlinear Aerodynamic Models," NATO AGARDOGRAPH, The Techniques and Technology of Nonlinear Filtering and Kalman Filtering, February 1982.
- 5 Bekley, G.A., Benken, J.E.W., "Identification of Biological Systems: A Survey," Automatica, Vol. 14, No. 1, Jan. 1978, pp 41-47.
- 6 Chow, G.C., "Identification and Estimation in Econometric Systems: A Survey," IEEE Trans. on Automatic Control, Vol. AC-19, No. 6, Dec. 1974, pp 855-861.
- 7 Gersch, W., and Foutch, D.A., "Least Squares Estimates of Structural System Parameters Using Covariance Function Data," IEEE Trans. on Automatic Control, Vol. AC-19, No. 6, Dec. 1974, pp 898-903.
- 8 Willsky, A.A., "A Survey of Design Methods for Failure Detection in Dynamic Systems," Automatica, Vol. 12, 1976, pp 601-610.
- 9 Gupta, N.K., and Hall, Jr., W.E., "Methods for the Real Time Identification of Vehicle Parameters," Technical Report No. 4 under Office of Naval Research Contract N00014-72-C-0328, Feb. 1976.
- 10 Gupta, N.K., Hall, Jr., W.E., "Input Design for Identification of Aircraft Stability and Control Derivatives," NASA Contractor Report CR-2493, Feb. 1975.

-
- 11 Gupta, N.K., Hall, Jr., W.E., "Design and Evaluation of Sensor Systems for State and Parameter Estimation," *J. Guidance and Control*, Vol. 1, No. 6, Nov. - Dec., 1978, pp 397-403.
 - 12 Gupta, N.K., Hall, Jr., W.E., Trankle, T.L., "Advanced Methods of Model Structure Determination from Test Data," *J. Guidance and Control*, Vol. 1, No. 3 May-June 1978, pp 197, 204.
 - 13 Allen, D.M., "Mean Square Error of Prediction as a Criterion for Selecting Variables," *Technometrics*, Vol. 13, No. 3, Aug. 1971, pp 469, 475.
 - 14 Lawson, C.L., and Hanson, R.J., *Solving Least Square Problems*, Prentice-Hall, 1974.
 - 15 Gupta, N.K. and Mehra, R.K., "Computational Aspects of Maximum Likelihood Estimation and Reduction in Sensitivity Function Calculations," *IEEE Trans. on Automatic Control*, Vol. AC-10, No. 6, Dec. 1974, pp 774, 783.
 - 16 Marquardt, D.W., "An Algorithm for Least Squares Estimation of Nonlinear Parameters," *J. Soc. Indust. Appl. Math.*, Vol. 11, No. 2, 1963, pp 431-441.
 - 17 Levenberg, K., "A Method for the Solution of Certain Nonlinear Problems in Least Squares," *Quarterly Applied Mathematics*, Vol. 2, 1944, pp. 164-168
 - 18 Marquardt, D. W., "An Algorithm for Least Squares Estimation of Nonlinear Parameters," *J. Soc. Indust. Appl. Math.*, Vol. 11, No. 2, 1963, pp 431-441
 - 19 Gupta, N.K. and Mehra, R.K., "Computational Aspects of Maximum Likelihood Estimation and Reduction in Sensitivity Function Calculations," *IEEE Trans. on Automatic Control*, Vol. AC-10, No. 6, Dec. 1974, pp 774, 783.
 - 20 Brown, K., M., Dennis, J. E., " Derivative Free Analogues for the Levenberg-Marquardt and Gauss Algorithms for Nonlinear Least Squares Approximation," *Numer. Math.*, Vol. 18, pp 289-297